Journal of Computational and Applied Mathematics 236 (2012) 4245–4263

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

A parareal waveform relaxation algorithm for semi-linear parabolic partial differential equations☆

Jun Liu [a,b], Yao-Lin Jiang [a,*]

[a] Department of Mathematical Sciences, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China

[b] College of Science, China University of Petroleum, Qingdao, Shandong 266580, China

ARTICLE INFO

Article history:
Received 8 August 2011
Received in revised form 22 February 2012

MSC:
65M06
65Y05

Keywords:
Parareal algorithm
Waveform relaxation
Hybrid parallelism
Convergence

ABSTRACT

We report a new parallel iterative algorithm for semi-linear parabolic partial differential equations (PDEs) by combining a kind of waveform relaxation (WR) techniques into the classical parareal algorithm. The parallelism can be simultaneously exploited by WR and parareal in different directions. We provide sharp error estimations for the new algorithm on bounded time domain and on unbounded time domain, respectively. The iterations of the parareal and the WR are balanced to optimize the performance of the algorithm. Furthermore, the speedup and the parallel efficiency of the new approach are analyzed. Numerical experiments are carried out to verify the effectiveness of the theoretic work.

1. Introduction

The waveform relaxation (WR) algorithm is an iterative algorithm to solve large scale systems of time-dependent equations in parallel. It is originally proposed to simulate large circuits [1], and has been widely applied for numerically solving ordinary differential equations (ODEs) and differential algebraic equations (DAEs); see [2–5]. Recently, the WR technique has been extended in [6] for solving semi-linear parabolic partial differential equations (PDEs) directly at the PDE level. Compared with the classical WR algorithms, the modified WR algorithm needs much less iterations for convergence, and the parallelism can be preserved.

The parallelism of the WR algorithm in [6] is tightly related to the iteration number for convergence, which means that it cannot exploit all the processors available on a given massively parallel computer. In this paper, we further consider the parallelism on the time domain for faster computation. In detail, we employ the parareal algorithm, together with the WR algorithm, to utilize both of the two different kinds of parallel fashions.

The parareal algorithm, which is introduced by Lions et al. in [7], is a time-parallel iterative algorithm for solving time-dependent differential equations. It has been clearly shown in [8] that, the so-called parareal algorithm is strictly related to the parallel-in-time method for ODEs defined in [9,10], based on a so-called parallel factorization. The parareal algorithm can be regarded as a variant of the multiple shooting method, or the multigrid-in-time algorithm; see [11–13]. The advantage of the parareal algorithm is that it allows the computation of the solution later in time, before having fully accurate approximations at earlier times, while the global accuracy of the iterative process after few iterations is comparable

☆ This work was supported by the Natural Science Foundation of China (NSFC) under grant 11071192, and the International Science and Technology Cooperation Program of China under grant 2010DFA14700.

* Corresponding author.
E-mail address: yljiang@mail.xjtu.edu.cn (Y.-L. Jiang).

0377-0427/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2012.05.014

to that given by a sequential numerical method used on a fine discretization in time [14]. At present, the parareal algorithm, as well as its variants, have been applied in financial mathematics [15], fluid mechanics [16], and quantum chemistry [17].

Taking the WR algorithm into the framework of parareal introduces several advantages. First, the new algorithm looks more flexible, and it can exploit more processors for fast computations. Second, by arranging the iteration number of WR during each parareal iteration, much computational cost can be saved.

The remainder of this paper is organized as follows. In Section 2, we propose the parareal WR algorithm. In Section 3, we give the error estimations for the parareal WR algorithm on bounded time domain and on unbounded time domain, respectively. In Section 4, the discrete parareal WR algorithm and its error estimation are presented. In Section 5, the balance of the two kinds of iterative processes is given to achieve better performance. The parallel efficiency of the hybrid algorithm is analyzed in Section 6. Numerical experiments are carried out in Section 7 to verify the effectiveness of our theoretic work.

2. Parareal WR algorithm

2.1. Recall of the parareal algorithm

We recall the parareal algorithm for the following system of ODEs,

$$\begin{cases} \dfrac{du(t)}{dt} = f(t, u(t)), \\ u(0) = u_0, \quad t \in [0, T], \end{cases} \tag{1}$$

where the nonlinear function $f : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ is Lipschitz on $\mathbb{R}^d$. First, the time domain $[0, T]$ is divided into N time slices $[T_n, T_{n+1}]$, for $n = 0, 1, \ldots, N-1$. We suppose that all the time slices are of uniform size, namely $\Delta T = \frac{T}{N}$. Then, on each time slice two propagators G and F are employed, where G is usually a low order numerical method, and F is usually of higher order or smaller time step. Therefore, G and F are named as the coarse and the fine propagator, respectively.

According to [7,11,13], the approximation at the time instant T_{n+1} can be updated by the iterative scheme

$$U_{n+1}^{k+1} = G(T_{n+1}, T_n, U_n^{k+1}) + F(T_{n+1}, T_n, U_n^k) - G(T_{n+1}, T_n, U_n^k), \tag{2}$$

with the initial approximation $U_{n+1}^0 = G(T_{n+1}, T_n, U_n^0)$, for $n = 0, 1, \ldots, N-1$. At the present iteration, the quantities U_n^k, for $n = 0, 1, \ldots, N-1$, are known; hence N processors can be employed to compute the N quantities $F(T_{n+1}, T_n, U_n^k)$ simultaneously. Generally, the parareal algorithm makes sense if the number of parareal iterations required to a desired accuracy is significantly less than N.

2.2. The construction of the parareal WR algorithm

We consider a semi-linear system of parabolic PDEs of the following form,

$$\begin{cases} \mathcal{L}u + c(\mathbf{x}, t)u = f(u, \mathbf{x}, t), \quad \mathbf{x} \in \Omega,\ 0 < t < T, \\ u(\mathbf{x}, t) = g(\mathbf{x}, t), \quad \mathbf{x} \in \partial\Omega,\ 0 \le t \le T, \\ u(\mathbf{x}, 0) = \varphi(\mathbf{x}), \quad \mathbf{x} \in \Omega, \end{cases} \tag{3}$$

where

$$\mathcal{L}u = \frac{\partial u}{\partial t} + L_t u, \quad L_t u = -\sum_{i,j=1}^{n} a_{i,j}(\mathbf{x}, t)\frac{\partial^2 u}{\partial \mathbf{x}_i \partial \mathbf{x}_j} + \sum_{i=1}^{n} b_i(\mathbf{x}, t)\frac{\partial u}{\partial \mathbf{x}_i},$$

and $a_{i,j}(\mathbf{x}, t), b_i(\mathbf{x}, t) \in C(\bar{\Omega} \times [0, T])$, f is a given nonlinear function. $-\mathcal{L}$ is a parabolic operator on $\Omega \times [0, T]$.

The WR technique proposed in [6], for system (3) on the time slice $[T_n, T_{n+1}]$ with initial condition $u(\mathbf{x}, T_n) = h(\mathbf{x})$, can be described as

$$\begin{cases} \mathcal{L}u^{(k+1)} + c(\mathbf{x}, t)u^{(k+1)} = \tilde{f}(u^{(k+1)}, u^{(k)}, \mathbf{x}, t), \quad \mathbf{x} \in \Omega,\ T_n < t < T_{n+1}, \\ u^{(k+1)}(\mathbf{x}, t) = g(\mathbf{x}, t), \quad \mathbf{x} \in \partial\Omega,\ T_n \le t \le T_{n+1}, \\ u^{(k+1)}(\mathbf{x}, T_n) = h(\mathbf{x}), \quad \mathbf{x} \in \Omega, \end{cases} \tag{4}$$

where the splitting function $\tilde{f}$ satisfies $\tilde{f}(u, u, \mathbf{x}, t) = f(u, \mathbf{x}, t)$, and the initial guess $u^{(0)}$ satisfies $u^{(0)}(\mathbf{x}, T_n) = h(\mathbf{x})$ for $\mathbf{x} \in \Omega$, and $u^{(0)}(\mathbf{x}, t) = g(\mathbf{x}, t)$ for $\mathbf{x} \in \partial\Omega$. The convergence and parallelism for the WR algorithm (4) can be found in [6]. We denote by $W_k(T_{n+1}, T_n, h(\mathbf{x}))$ the approximation at T_{n+1}, after k iterations of WR by scheme (4). Replacing the F propagator in the classical parareal algorithm by the W_k propagator, we can obtain a modified parareal algorithm, say parareal WR algorithm. Its iterative scheme is

$$\begin{cases} U_0^{k+1}(\mathbf{x}) = \varphi(\mathbf{x}), \\ U_{n+1}^{k+1}(\mathbf{x}) = G(T_{n+1}, T_n, U_n^{k+1}(\mathbf{x})) + W_{k+1}(T_{n+1}, T_n, U_n^k(\mathbf{x})) - G(T_{n+1}, T_n, U_n^k(\mathbf{x})), \end{cases} \tag{5}$$

where the WR iteration number is assigned to coincide with the index of the parareal WR algorithm. It means that, $(k+1)$ iterations of WR are performed on each time slice to create $W_{k+1}(T_{n+1}, T_n, U_n^k(\mathbf{x}))$ for the $(k+1)$th parareal iteration.

In the remainder of this paper, we will analyze the convergence of the parareal WR algorithm for the following one-dimensional semi-linear parabolic PDE with homogeneous boundary conditions

$$\begin{cases} \dfrac{\partial u}{\partial t} - \dfrac{\partial^2 u}{\partial x^2} = f(u), \quad 0 < x < l,\ 0 < t < T, \\ u(0,t) = u(l,t) = 0, \quad 0 \le t \le T, \\ u(x,0) = \varphi(x), \quad 0 \le x \le l, \end{cases} \tag{6}$$

where the function $\varphi : [0, l] \to \mathbb{R}$, and the nonlinear function f is Lipschitz on $\mathbb{R}$. Next, we show respectively the properties of the coarse and the fine propagators.

2.3. The properties of the coarse propagator

We use the following semi-implicit scheme for system (6) on time slice $[T_n, T_{n+1}]$, to generate the coarse propagator,

$$\begin{cases} \dfrac{U_{m+1}(x) - U_m(x)}{\Delta t} - \dfrac{d^2 U_{m+1}(x)}{dx^2} = f(U_m(x)), \quad m = 0, 1, \ldots, q-1, \\ U_0(x) = h(x), \quad 0 \le x \le l, \end{cases} \tag{7}$$

where $U_m(x)$ is an approximation to $u(x, T_n + m\Delta t)$, and $\Delta T = q\Delta t$. In fact, the formula in (7) provides a time-stepping scheme. After m steps, $U_m(x)$ is a known function, and $U_{m+1}(x)$ can be regarded as the solution of the following linear ordinary differential equation of second order,

$$\begin{cases} \dfrac{d^2 U_{m+1}(x)}{dx^2} - \dfrac{1}{\Delta t} U_{m+1}(x) + \dfrac{1}{\Delta t} U_m(x) + f(U_m(x)) = 0, \quad 0 \le x \le l, \\ U_{m+1}(0) = U_{m+1}(l) = 0. \end{cases} \tag{8}$$

The characteristic equation of the associated homogeneous differential equation of system (8) is

$$\lambda^2 = \frac{1}{\Delta t},$$

from which we can find that the complementary function of system (8) is

$$U_{m+1,c}(x) = \theta_1 e^{\frac{x}{\sqrt{\Delta t}}} + \theta_2 e^{-\frac{x}{\sqrt{\Delta t}}},$$

where θ_1 and θ_2 are constants. We can seek a solution of system (8) by variation of parameters. Suppose a solution of system (8) is in the form

$$U_{m+1}(x) = \theta_1(x) e^{\frac{x}{\sqrt{\Delta t}}} + \theta_2(x) e^{-\frac{x}{\sqrt{\Delta t}}},$$

where $\theta_1(x)$ and $\theta_2(x)$ are two unknown functions. Substitute this formula into system (8), we can obtain

$$\begin{cases} \dot\theta_1(x) e^{\frac{x}{\sqrt{\Delta t}}} + \dot\theta_2(x) e^{-\frac{x}{\sqrt{\Delta t}}} = 0, \\ \dot\theta_1(x) \dfrac{1}{\sqrt{\Delta t}} e^{\frac{x}{\sqrt{\Delta t}}} - \dot\theta_2(x) \dfrac{1}{\sqrt{\Delta t}} e^{-\frac{x}{\sqrt{\Delta t}}} = -\dfrac{1}{\Delta t} U_m(x) - f(U_m(x)), \end{cases}$$

where $\dot\theta_i(x)$ denotes the derivative of $\theta_i(x)$ with respect to x, and $i = 1, 2$. The expressions for $\theta_1(x)$ and $\theta_2(x)$ are

$$\theta_1(x) = -\int_0^x \frac{\sqrt{\Delta t}}{2} e^{-\frac{s}{\sqrt{\Delta t}}} \left(\frac{1}{\Delta t} U_m(s) + f(U_m(s)) \right) ds + \Theta_1,$$

$$\theta_2(x) = \int_0^x \frac{\sqrt{\Delta t}}{2} e^{\frac{s}{\sqrt{\Delta t}}} \left(\frac{1}{\Delta t} U_m(s) + f(U_m(s)) \right) ds + \Theta_2,$$

where Θ_1 and Θ_2 are constants. The solution for system (8) can be expressed as

$$U_{m+1}(x) = -\int_0^x \frac{\sqrt{\Delta t}}{2} (e^{\frac{x-s}{\sqrt{\Delta t}}} - e^{-\frac{x-s}{\sqrt{\Delta t}}}) \left(\frac{1}{\Delta t} U_m(s) + f(U_m(s)) \right) ds + \Theta_1 e^{\frac{x}{\sqrt{\Delta t}}} + \Theta_2 e^{-\frac{x}{\sqrt{\Delta t}}}.$$

Together with the initial conditions of system (8), we have

$$U_{m+1}(x) = \bar{C}_1 F(x) - \int_0^x F(x-s) \left[\frac{1}{\Delta t} U_m(s) + f(U_m(s)) \right] ds, \tag{9}$$

where

$$\bar{C}_1 = \int_0^l \frac{e^{\frac{l-s}{\sqrt{\Delta t}}} - e^{-\frac{l-s}{\sqrt{\Delta t}}}}{e^{\frac{l}{\sqrt{\Delta t}}} - e^{-\frac{l}{\sqrt{\Delta t}}}} \left[\frac{1}{\Delta t} U_m(s) + f(U_m(s))\right] ds, \qquad F(x) = \frac{\sqrt{\Delta t}}{2}(e^{\frac{x}{\sqrt{\Delta t}}} - e^{-\frac{x}{\sqrt{\Delta t}}}).$$

For simplicity we denote the function

$$\phi(x) = e^{\frac{x}{\sqrt{\Delta t}}} - e^{-\frac{x}{\sqrt{\Delta t}}}, \quad 0 \le x \le l.$$

We further denote $U_{m+1}(x) = \tilde{G}(U_m(x))$, then $G(T_{n+1}, T_n, U_n(x)) = \tilde{G}^q(U_n(x))$, and the G propagator has the following two properties.

Lemma 2.1. *We assume that the derivative of the nonlinear function f in system (6) is bounded by a positive constant M, then there exists a positive constant C_2 such that the coarse propagator G satisfies*

$$\max_{0\le x\le l} |G(T_{n+1}, T_n, U(x)) - G(T_{n+1}, T_n, V(x))| \le (1 + C_2\Delta T) \max_{0\le x\le l} |U(x) - V(x)|,$$

where $U(x)$ and $V(x)$ are continuously differential functions, and satisfy $U(0) = U(l) = 0$, $V(0) = V(l) = 0$.

Proof. We only prove the result for the case $q = 1$, namely $G = \tilde{G}$. According to the solution for system (7), we have

$$\tilde{G}(V(x)) = \bar{C}_2 F(x) - \int_0^x F(x-s)\left[\frac{1}{\Delta t} V(s) + f(V(s))\right] ds,$$

where

$$\bar{C}_2 = \int_0^l \frac{\phi(l-s)}{\phi(l)}\left[\frac{1}{\Delta t} V(s) + f(V(s))\right] ds.$$

Then,

$$\begin{aligned}\tilde{G}(U(x)) - \tilde{G}(V(x)) = {} & F(x)\int_0^l \frac{\phi(l-s)}{\phi(l)}\left[\frac{1}{\Delta t}(U(s) - V(s)) + f(U(s)) - f(V(s))\right] ds \\ & - F(x)\int_0^x \frac{\phi(x-s)}{\phi(x)}\left[\frac{1}{\Delta t}(U(s) - V(s)) + f(U(s)) - f(V(s))\right] ds.\end{aligned}$$

We notice that $(e^{x-s} - e^{s-x})/(e^x - e^{-x})$ is a monotonically increasing function with respect to x, and for any $0 \le s \le x$,

$$\frac{\phi(l-s)}{\phi(l)} - \frac{\phi(x-s)}{\phi(x)} \ge 0.$$

Then we have,

$$\begin{aligned}|\tilde{G}(U(x)) - \tilde{G}(V(x))| \le {} & \int_0^x \left(\frac{\phi(l-s)}{\phi(l)} - \frac{\phi(x-s)}{\phi(x)}\right) ds \cdot F(x)\left(\frac{1}{\Delta t} + M\right) \max_{0\le x\le l} |U(x) - V(x)| \\ & + \int_x^l \frac{\phi(l-s)}{\phi(l)} ds \cdot F(x)\left(\frac{1}{\Delta t} + M\right) \max_{0\le x\le l} |U(x) - V(x)| \\ = {} & \left[\frac{\sqrt{\Delta t}(\phi(l) - 2)}{\phi(l)} - \frac{\sqrt{\Delta t}(\phi(x) - 2)}{\phi(x)}\right] \frac{\sqrt{\Delta t}}{2}\phi(x)\left(\frac{1}{\Delta t} + M\right) \max_{0\le x\le l} |U(x) - V(x)| \\ = {} & \left[1 - \frac{\phi(x)}{\phi(l)} - \frac{\phi(l-x)}{\phi(l)}\right](1 + M\Delta t) \max_{0\le x\le l} |U(x) - V(x)|.\end{aligned}$$

Comparing the corresponding items in the Taylor expansions for the numerator $\phi(x) + \phi(l-x)$ and the denominator $\phi(l)$, we know that

$$0 < 1 - \frac{\phi(x)}{\phi(l)} - \frac{\phi(l-x)}{\phi(l)} \le 1,$$

which leads to

$$|\tilde{G}(U(x)) - \tilde{G}(V(x))| < (1 + M\Delta t) \max_{0\le x\le l} |U(x) - V(x)|.$$

The result follows. □

We notice that, the constant C_2 in Lemma 2.1 can be chosen as M for the case $q = 1$, while in the case $q > 1$, the constant C_2 is usually bigger than M. For example, if $q = 2$,

$$\begin{aligned}\max_{0\le x\le l}|G(T_{n+1},T_n,U(x))-G(T_{n+1},T_n,V(x))| &= \max_{0\le x\le l}|\tilde{G}(\tilde{G}(U(x)))-\tilde{G}(\tilde{G}(V(x)))|\\ &\le \left(1+M\frac{\Delta T}{2}\right)\max_{0\le x\le l}|\tilde{G}(U(x))-\tilde{G}(V(x))|\\ &\le \left(1+M\frac{\Delta T}{2}\right)^2\max_{0\le x\le l}|U(x)-V(x)|.\end{aligned}$$

There exists a constant C_2, such that

$$\left(1+M\frac{\Delta T}{2}\right)^2 \le 1+C_2\Delta T,$$

for small ΔT. Likewise, the result in Lemma 2.1 also holds for the case $q > 2$.

The truncation error of the coarse propagator can be given by the following lemma.

Lemma 2.2. *When using scheme (7) for system (6), on the time slice $[T_n, T_{n+1}]$ with the initial condition $u(x, T_n) = h(x)$, we have*

$$S(T_{n+1},T_n,h(x))-G(T_{n+1},T_n,h(x)) = c_2(h(x))\Delta T^2+c_3(h(x))\Delta T^3+\cdots,$$

where $S(T_{n+1}, T_n, h(x))$ denotes the true solution at T_{n+1} with homogeneous boundary conditions and the initial condition $u(T_n) = h(x)$, and $c_j(h(x))$ are continuously differentiable for $j = 2, 3, \ldots$.

Proof. The result for the case $q = 1$ is obvious, and we only prove the result for the case $q = 2$. The quantity $S(T_{n+1}, T_n, h(x))$ can be expanded as

$$\begin{aligned}S(T_{n+1},T_n,h(x)) &= h(x)+\frac{\partial u(x,T_n)}{\partial t}\Delta T+\frac{1}{2!}\frac{\partial^2 u(x,T_n)}{\partial t^2}\Delta T^2+O(\Delta T^3)\\ &= h(x)+\left[\frac{\partial^2 u(x,T_n)}{\partial x^2}+f(u(x,T_n))\right]\Delta T+\frac{1}{2!}\frac{\partial^2 u(x,T_n)}{\partial t^2}\Delta T^2+O(\Delta T^3),\end{aligned}$$

and the quantity $G(T_{n+1}, T_n, h(x))$ can be written as

$$\begin{aligned}G(T_{n+1},T_n,h(x)) = \tilde{G}(\tilde{G}(h(x))) &= \tilde{G}(h(x))+\frac{\Delta T}{2}\left[\frac{\partial^2\tilde{G}(\tilde{G}(h(x)))}{\partial x^2}+f(\tilde{G}(h(x)))\right]\\ &= h(x)+\frac{\Delta T}{2}\left[\frac{\partial^2\tilde{G}(h(x))}{\partial x^2}+f(h(x))+\frac{\partial^2\tilde{G}(\tilde{G}(h(x)))}{\partial x^2}+f(\tilde{G}(h(x)))\right].\end{aligned}$$

It is easy to check that

$$\begin{aligned}&\frac{\partial^2 u(x,T_n)}{\partial x^2}-\frac{1}{2}\frac{\partial^2\tilde{G}(h(x))}{\partial x^2}-\frac{1}{2}\frac{\partial^2\tilde{G}(\tilde{G}(h(x)))}{\partial x^2} = \frac{\partial^2 u(x,T_n)}{\partial x^2}-\frac{\partial^2\tilde{G}(h(x))}{\partial x^2}+\frac{1}{2}\frac{\partial^2\tilde{G}(h(x))}{\partial x^2}-\frac{1}{2}\frac{\partial^2\tilde{G}(\tilde{G}(h(x)))}{\partial x^2}\\ &= -\frac{\Delta T}{2}\left[\frac{\partial^4\tilde{G}(h(x))}{\partial x^4}+\frac{\partial^2 f(h(x))}{\partial x^2}\right]-\frac{1}{2}\frac{\Delta T}{2}\left[\frac{\partial^4\tilde{G}(\tilde{G}(h(x)))}{\partial x^4}+\frac{\partial^2 f(\tilde{G}(h(x)))}{\partial x^2}\right]\end{aligned}$$

and

$$\begin{aligned}f(u(x,T_n))-\frac{1}{2}f(h(x))-\frac{1}{2}f(\tilde{G}(h(x))) &= \frac{1}{2}f(h(x))-\frac{1}{2}f(\tilde{G}(h(x)))\\ &= -\frac{1}{2}[f'(\xi(x))(\tilde{G}(h(x))-h(x))] = -\frac{1}{2}f'(\xi(x))\frac{\Delta T}{2}\left(\frac{\partial^2\tilde{G}(h(x))}{\partial x^2}+f(h(x))\right).\end{aligned}$$

Then

$$S(T_{n+1},T_n,h(x))-G(T_{n+1},T_n,h(x)) = c_2(h(x))\Delta T^2+c_3(h(x))\Delta T^3+\cdots,$$

which completes the proof. □

Obviously, there exists a constant C_3, such that the local truncation error of G is bounded by $C_3\Delta T^2$ for small ΔT.

2.4. The properties of the fine propagator

For simplicity, we consider the following WR technique for system (6) on $[T_n, T_{n+1}]$,

$$\begin{cases} \dfrac{\partial u^{(i+1)}}{\partial t} - \dfrac{\partial^2 u^{(i+1)}}{\partial x^2} = f(u^{(i)}), \quad 0 < x < l,\ T_n < t < T_{n+1}, \\ u^{(i+1)}(0,t) = u^{(i+1)}(l,t) = 0, \quad T_n \le t \le T_{n+1}, \\ u^{(i+1)}(x, T_n) = h(x), \quad 0 \le x \le l. \end{cases} \tag{10}$$

The initial guess is chosen as $u^{(0)}(x,t) = h(x)$, where $t \in [T_n, T_{n+1}]$. For any fixed i, system (10) is a linear system, with the solution

$$u^{(i+1)}(x,t) = \sum_{n=1}^{\infty} a_n^{(i+1)}(t) \sin\frac{n\pi x}{l},$$

where

$$a_n^{(i+1)}(t) = \frac{2}{l}\int_0^l h(x)\sin\frac{n\pi x}{l}dx e^{-(\frac{n\pi}{l})^2(t-T_n)} + \frac{2}{l}\int_{T_n}^t\int_0^l f(u^{(i)}(\xi,\tau))\sin\frac{n\pi\xi}{l}d\xi e^{-(\frac{n\pi}{l})^2(t-\tau)}d\tau. \tag{11}$$

We define the function

$$a_n(t) = \frac{2}{l}\int_0^l h(x)\sin\frac{n\pi x}{l}dx e^{-(\frac{n\pi}{l})^2(t-T_n)} + \frac{2}{l}\int_{T_n}^t\int_0^l f(u(\xi,\tau))\sin\frac{n\pi\xi}{l}d\xi\ e^{-(\frac{n\pi}{l})^2(t-\tau)}d\tau, \tag{12}$$

and it is easy to check that the function $\sum_{n=1}^{\infty} a_n(t)\sin\frac{n\pi x}{l}$ is the solution for system (6). The difference between $u^{(i)}(x,t)$ and $u(x,t)$ can be estimated by the following theorem, and the proof can be found in [6].

Theorem 2.1. *For system (6) defined on $[T_n, T_{n+1}]$ with an initial function $h(x)$ at T_n, we assume that the derivative of the nonlinear function f is uniformly bounded by a constant M, then the sequence of $\{u^{(i)}(x,t)\}$ defined by scheme (10) converges to the true solution $u(x,t)$ of system (6), and satisfies*

$$\max_{0\le x\le l, T_n\le t\le T_{n+1}} |u^{(i)}(x,t) - u(x,t)| \le \frac{(\tilde{C}\Delta T)^i}{i!} \max_{0\le x\le l, T_n\le t\le T_{n+1}} |u^{(0)}(x,t) - u(x,t)|,$$

where $\tilde{C}$ is a constant. Furthermore, we have

$$\max_{0\le x\le l, T_n\le t\le T_{n+1}} |u^{(0)}(x,t) - u(x,t)| = \max_{0\le x\le l, T_n\le t\le T_{n+1}} |h(x) - u(x,t)| \le \bar{C}\Delta T,$$

where $\bar{C}$ is a constant, then,

$$\max_{0\le x\le l, T_n\le t\le T_{n+1}} |u^{(i)}(x,t) - u(x,t)| \le \bar{C}\frac{(\tilde{C}\Delta T)^i}{i!}\Delta T,$$

which means that the WR algorithm (10) converges superlinearly.

3. Convergence analysis

In this section, we derive the convergence results for the parareal WR algorithm on bounded time domain and on unbounded time domain, respectively.

3.1. Approximate superlinear convergence on bounded time domain

Similar to the convergence theorem provided in [18] for the classical parareal algorithm, the convergence result for the parareal WR algorithm on bounded time domain is given by the following theorem.

Theorem 3.1. *For system (6) defined on time domain $[0, T]$, we assume that the derivative of the nonlinear function f is bounded by a constant M, then the parareal WR algorithm (5) converges, and the error is bounded by*

$$\max_{0\le x\le l} |u(x,T_n) - U_n^k(x)| \le \frac{C_3 e^{C_2 T_{n-k-1}}}{C_1}\frac{(C_1 T_n \Delta T)^{k+1}}{(k+1)!} + \bar{C}\Delta T^k \sum_{i=0}^{k-1} e^{C_2 T_{n-i-1}}\frac{\tilde{C}^{k-i}}{(k-i)!}\frac{C_1^i T_n^{i+1}}{(i+1)!}, \tag{13}$$

where constants C_1, C_2, C_3, $\tilde{C}$ and $\bar{C}$ are defined in Lemmas 2.1 and 2.2 and Theorem 2.1, respectively.

Proof. The error of the parareal WR algorithm at T_{n+1} can be expressed as

$$\begin{aligned} u(x,T_{n+1})-U_{n+1}^{k+1}(x) &= S(T_{n+1},T_n,u(x,T_n))-G(T_{n+1},T_n,U_n^{k+1}(x))-W_{k+1}(T_{n+1},T_n,U_n^k(x))+G(T_{n+1},T_n,U_n^k(x)) \\ &= [S(T_{n+1},T_n,u(x,T_n))-G(T_{n+1},T_n,u(x,T_n))+G(T_{n+1},T_n,U_n^k(x))-S(T_{n+1},T_n,U_n^k(x))] \\ &\quad +[S(T_{n+1},T_n,U_n^k(x))-W_{k+1}(T_{n+1},T_n,U_n^k(x))] \\ &\quad +[G(T_{n+1},T_n,u(x,T_n))-G(T_{n+1},T_n,U_n^{k+1}(x))]. \end{aligned}$$

From Lemmas 2.1 and 2.2, for small ΔT,

$$\begin{aligned} \max_{0\le x\le l}|u(x,T_{n+1})-U_{n+1}^{k+1}(x)| &\le C_1\Delta T^2\max_{0\le x\le l}|u(x,T_n)-U_n^k(x)|+(1+C_2\Delta T)\max_{0\le x\le l}|u(x,T_n)-U_n^{k+1}(x)| \\ &\quad +\max_{0\le x\le l}|S(T_{n+1},T_n,U_n^k(x))-W_{k+1}(T_{n+1},T_n,U_n^k(x))|. \end{aligned}$$

Similarly, we have

$$\begin{aligned} \max_{0\le x\le l}|u(x,T_{n+1})-U_{n+1}^{0}(x)| &\le \max_{0\le x\le l}|S(T_{n+1},T_n,u(x,T_n))-G(T_{n+1},T_n,u(x,T_n))| \\ &\quad +\max_{0\le x\le l}|G(T_{n+1},T_n,u(x,T_n))-G(T_{n+1},T_n,U_n^0(x))| \\ &\le C_3\Delta T^2+(1+C_2\Delta T)\max_{0\le x\le l}|u(x,T_n)-U_n^0(x)|. \end{aligned}$$

Now, we study the recurrence relations

$$e_{n+1}^{k+1}=\alpha e_n^k+\beta e_n^{k+1}+\epsilon(k+1)\Delta T,\qquad e_{n+1}^0=\gamma+\beta e_n^0, \tag{14}$$

where $\alpha=C_1\Delta T^2$, $\beta=1+C_2\Delta T$, $\gamma=C_3\Delta T^2$, and $\epsilon(k)=\bar{C}\frac{(\tilde{C}\Delta T)^k}{k!}$, with $e_0^k=0$ for any $k\ge 0$. It is easy to check that, e_n^k is an upper bound on $\max_{0\le x\le l}|u(x,T_n)-U_n^k(x)|$. In fact, if we suppose e_i^j is an upper bound on $\max_{0\le x\le l}|u(x,T_i)-U_i^j(x)|$, for $j=0,1,\ldots,k+1$ and $i=0,1,\ldots,n$, then

$$\begin{aligned} e_{n+1}^{k+1} &= \alpha e_n^k+\beta e_n^{k+1}+\epsilon(k+1)\Delta T \\ &\ge \alpha\max_{0\le x\le l}|u(x,T_n)-U_n^k(x)|+\beta\max_{0\le x\le l}|u(x,T_n)-U_n^{k+1}(x)|+\max_{0\le x\le l}|S(T_{n+1},T_n,U_n^k(x))-W_{k+1}(T_{n+1},T_n,U_n^k(x))| \\ &\ge \max_{0\le x\le l}|u(x,T_{n+1})-U_{n+1}^{k+1}(x)|. \end{aligned}$$

By induction, e_n^k is an upper bound on $\max_{0\le x\le l}|u(x,T_n)-U_n^k(x)|$.

Multiplying (14) by ζ^{n+1} and summing over n, we obtain

$$\sum_{n\ge 0}e_{n+1}^{k+1}\zeta^{n+1}=\alpha\sum_{n\ge 0}e_n^k\zeta^{n+1}+\beta\sum_{n\ge 0}e_n^{k+1}\zeta^{n+1}+\sum_{n\ge 0}\epsilon(k+1)\Delta T\zeta^{n+1},$$

and

$$\sum_{n\ge 0}e_{n+1}^{0}\zeta^{n+1}=\sum_{n\ge 0}\gamma\zeta^{n+1}+\sum_{n\ge 0}\beta e_n^0\zeta^{n+1},$$

where $0<\zeta<\frac{1}{\beta}<1$. We define the functions $\rho_k(\zeta):=\sum_{n\ge 0}e_{n+1}^k\zeta^{n+1}$, for $k=0,1,\ldots$, which satisfy the relations

$$\rho_{k+1}(\zeta)=\alpha\zeta\rho_k(\zeta)+\beta\zeta\rho_{k+1}(\zeta)+\epsilon(k+1)\Delta T\frac{\zeta}{1-\zeta},\qquad \rho_0(\zeta)=\frac{\gamma\zeta}{(1-\zeta)(1-\beta\zeta)}.$$

Solving for $\rho_{k+1}(\zeta)$, we have

$$\rho_{k+1}(\zeta)=\frac{\alpha\zeta}{1-\beta\zeta}\rho_k(\zeta)+\epsilon(k+1)\Delta T\frac{\zeta}{(1-\zeta)(1-\beta\zeta)}. \tag{15}$$

For simplicity, we denote

$$a(\zeta)=\frac{\alpha\zeta}{(1-\beta\zeta)},\qquad b(\zeta)=\frac{\zeta}{(1-\zeta)(1-\beta\zeta)}.$$

After induction, we have

$$\begin{aligned}\rho_k(\zeta) &= a(\zeta)^k\rho_0(\zeta) + \sum_{i=0}^{k-1}\epsilon(k-i)\Delta T a(\zeta)^i b(\zeta)\\ &= \frac{(\alpha\zeta)^k}{(1-\beta\zeta)^k}\frac{\gamma\zeta}{(1-\zeta)(1-\beta\zeta)} + \sum_{i=0}^{k-1}\epsilon(k-i)\Delta T\frac{(\alpha\zeta)^i}{(1-\beta\zeta)^i}\frac{\zeta}{(1-\zeta)(1-\beta\zeta)}.\end{aligned}$$

Replacing the factor $(1-\zeta)$ in the denominator by $(1-\beta\zeta)$ will increase the coefficients in the power series of $\rho_k(\zeta)$. It means that, the coefficients in the power series of $\rho_k(\zeta)$ are smaller than the corresponding coefficients in the power series of $\tilde{\rho}_k(\zeta)$, with

$$\tilde{\rho}_k(\zeta) = \frac{\gamma\alpha^k\zeta^{k+1}}{(1-\beta\zeta)^{k+2}} + \sum_{i=0}^{k-1}\epsilon(k-i)\Delta T\frac{\alpha^i\zeta^{i+1}}{(1-\beta\zeta)^{i+2}}.$$

Using the binomial series expansion

$$\frac{1}{(1-\beta\zeta)^{k+2}} = \sum_{j\geq 0}\binom{k+1+j}{j}\beta^j\zeta^j,$$

we can obtain the coefficients of the term ζ^n in the power series of $\tilde{\rho}_k(\zeta)$

$$\begin{aligned}\tilde{c}_n^k &= \gamma\alpha^k\binom{n}{k+1}\beta^{n-k-1} + \sum_{i=0}^{k-1}\epsilon(k-i)\Delta T\alpha^i\binom{n}{i+1}\beta^{n-i-1}\\ &\leq C_3C_1^k\Delta T^{2k+2}\frac{n(n-1)\cdots(n-k)}{(k+1)!}(1+C_2\Delta T)^{n-k-1}\\ &\quad + \bar{C}\sum_{i=0}^{k-1}\frac{(\tilde{C}\Delta T)^{k-i}}{(k-i)!}C_1^i\Delta T^{2i+1}\frac{n(n-1)\cdots(n-i)}{(i+1)!}(1+C_2\Delta T)^{n-i-1}\\ &\leq \frac{C_3e^{C_2T_{n-k-1}}}{C_1}\frac{(C_1T_n\Delta T)^{k+1}}{(k+1)!} + \bar{C}\Delta T^k\sum_{i=0}^{k-1}e^{C_2T_{n-i-1}}\frac{\tilde{C}^{k-i}}{(k-i)!}\frac{C_1^iT_n^{i+1}}{(i+1)!}.\end{aligned}$$

The quantity $\tilde{c}_n^k$ is an upper bound for e_n^k. Then the result follows. □

We can see that, each component in the right hand side of inequality (13) approaches zero superlinearly as k increases, even though more components arise. We conclude that the parareal WR algorithm (5) converges with approximate superlinear rate.

3.2. Convergence on unbounded time domain

The convergence for the parareal WR algorithm on unbounded time domain is discussed in the following theorem.

Theorem 3.2. *For system (6) defined on an unbounded time domain, we assume the derivative of the nonlinear function f to be bounded by a constant M, then the parareal WR algorithm (5) converges. That is*

$$\lim_{k\to\infty}\max_{0\leq x\leq l}|u(x,T_n) - U_n^k(x)| = 0, \quad n = 0, 1, \ldots.$$

Proof. Following the route in the proof of Theorem 3.1, we can also obtain the recurrence relation (15) on $\rho_k(\zeta)$, where ζ is restricted in $[0, \frac{1}{2(\alpha+\beta)}]$. For any small $\varepsilon > 0$, there exists an integer k_0, such that $\epsilon(k) < \frac{\varepsilon}{8\Delta T}$ for any $k > k_0$. Solving for $\rho_k(\zeta)$, we obtain

$$\begin{aligned}\rho_k(\zeta) &= a(\zeta)^{k-k_0}\rho_{k_0}(\zeta) + \sum_{i=0}^{k-k_0-1}\epsilon(k-i)\Delta T a(\zeta)^i b(\zeta)\\ &\leq a(\zeta)^{k-k_0}\rho_{k_0}(\zeta) + \sum_{i=0}^{k-k_0-1}\frac{\varepsilon}{8}a(\zeta)^i b(\zeta)\\ &= a(\zeta)^{k-k_0}\rho_{k_0}(\zeta) + \frac{1-a(\zeta)^{k-k_0}}{1-a(\zeta)}b(\zeta)\frac{\varepsilon}{8},\end{aligned}$$

where $a(\zeta) = \frac{\alpha\zeta}{1-\beta\zeta}$, and $b(\zeta) = \frac{\zeta}{(1-\zeta)(1-\beta\zeta)}$. In fact, both $a(\zeta)$ and $b(\zeta)$ are monotonically increasing functions on $[0, \frac{1}{2(\alpha+\beta)}]$, and

$$a(\zeta) = \frac{\alpha\zeta}{1-\beta\zeta} \le \frac{\frac{\alpha}{2(\alpha+\beta)}}{1-\frac{\beta}{2(\alpha+\beta)}} = \frac{\alpha}{2\alpha+\beta} < \frac{\alpha}{2\alpha} = \frac{1}{2},$$

$$b(\zeta) \le \frac{\frac{1}{2(\alpha+\beta)}}{(1-\frac{1}{2(\alpha+\beta)})(1-\frac{\beta}{2(\alpha+\beta)})} = \frac{2(\alpha+\beta)}{(2(\alpha+\beta)-1)(2\alpha+\beta)} < \frac{2}{2(\alpha+\beta)-1} < 2.$$

Moreover, the first k_0 iterations of the parareal WR algorithm can be regarded as k_0 corrections. Therefore, it is reasonable to assume that $\rho_{k_0}(\zeta) \le \rho_0(\zeta)$. From (14) we have

$$e_n^0 = \gamma(1+\beta+\cdots+\beta^{n-1}) = \gamma\frac{\beta^n-1}{\beta-1},$$

$$\rho_0(\zeta) = \sum_{n\ge0} e_n^0\zeta^n = \sum_{n\ge0}\gamma\frac{\beta^n-1}{\beta-1}\zeta^n \le \sum_{n\ge0}\frac{\gamma}{\beta-1}\frac{\beta^n-1}{2^n(\alpha+\beta)^n} \le \sum_{n\ge0}\frac{\gamma}{\beta-1}\frac{\beta^n-1}{2^n\beta^n} \le \frac{2\gamma}{\beta-1}.$$

In fact, there exists an integer k_1, such that $a(\zeta)^{k-k_0} < \frac{\beta-1}{4\gamma}\varepsilon$ for any $k-k_0 > k_1$. Therefore, for any $k > k_0+k_1$, we have

$$\rho_k(\zeta) \le \frac{\beta-1}{4\gamma}\varepsilon\rho_0(\zeta) + \frac{1}{1-a(\zeta)}b(\zeta)\frac{\varepsilon}{8} < \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon,$$

which means the power series of $\rho_k(\zeta)$ converge uniformly to 0. The corresponding coefficients also converge to 0, and the result follows. □

4. Discrete parareal WR algorithm

To perform the parareal WR algorithm numerically for system (6), we have to discretize both x and t variables. Before analyzing the discrete parareal WR algorithm, we employ the following notations.

- The quantity U_n^k denotes the approximate vector at the time instant T_n in the kth parareal iteration. Its entry $U_{n,j}^k$ is an approximation to $u(j\Delta x, T_n)$, where $j = 1, 2, \ldots, J-1$, and Δx is the step size of the equispaced space grid, with $J\Delta x = l$.
- The quantity $V_m^{(i)}$, which is associated with the fine propagator, denotes the approximate vector at the time instant $T_n + m\delta t$, by i iterations of WR on $[T_n, T_{n+1}]$, with the initial value $V_0^{(i)} = U_n^k$, where $i = 1, 2, \ldots, k$, $m = 1, \ldots, \bar{q}$, and $\bar{q}\delta t = \Delta T$. Its entry $V_{m,j}^{(i)}$ is an approximation to $u(j\Delta x, T_n + m\delta t)$. The discrete scheme for the WR propagator is

$$\frac{V_{m,j}^{(i+1)} - V_{m-1,j}^{(i+1)}}{\delta t} - \frac{V_{m,j+1}^{(i+1)} - 2V_{m,j}^{(i+1)} + V_{m,j-1}^{(i+1)}}{\Delta x^2} = f(V_{m,j}^{(i)}), \tag{16}$$

and the initial guess is $V_m^{(0)} = U_n^k$, for $m = 1, \ldots, \bar{q}$. The corresponding discrete scheme for system (6) is

$$\frac{V_{m,j} - V_{m-1,j}}{\delta t} - \frac{V_{m,j+1} - 2V_{m,j} + V_{m,j-1}}{\Delta x^2} = f(V_{m,j}). \tag{17}$$

- The quantity W_p, which is associated with the coarse propagator, denotes the approximate vector at the time instant $T_n + p\Delta t$ on $[T_n, T_{n+1}]$, with the initial value $W_0 = U_n^{k+1}$, where $p = 1, \ldots, q$, and $q\Delta t = \Delta T$. Its entry $W_{p,j}$ is an approximation to $u(j\Delta x, T_n + p\Delta t)$. The discrete scheme for the coarse propagator is

$$\frac{W_{p,j} - W_{p-1,j}}{\Delta t} - \frac{W_{p,j+1} - 2W_{p,j} + W_{p,j-1}}{\Delta x^2} = f(W_{p-1,j}). \tag{18}$$

Using above notations, we can obtain the scheme for the discrete parareal WR algorithm as follows:

$$\begin{cases} U_0^{k+1} = [u(\Delta x, 0), u(2\Delta x, 0), \ldots, u((J-1)\Delta x, 0)]^{\mathrm{T}} \\ U_{n+1}^{k+1} = G(T_{n+1}, T_n, U_n^{k+1}) + DW_{k+1}(T_{n+1}, T_n, U_n^k) - G(T_{n+1}, T_n, U_n^k). \end{cases} \tag{19}$$

In scheme (19), the quantity $G(T_{n+1}, T_n, U_n^{k+1})$ can be obtained from scheme (18) by setting

$$W_0 = U_n^{k+1}, \qquad G(T_{n+1}, T_n, U_n^{k+1}) = W_q.$$

The quantity $DW_{k+1}(T_{n+1}, T_n, U_n^k)$ can be obtained from scheme (16) by setting

$$V_0 = U_n^k, \qquad DW_{k+1}(T_{n+1}, T_n, U_n^k) = V_{\bar{q}}^{(k+1)}.$$

Next, we present two lemmas to show the properties for the coarse and the fine propagator, respectively.

Lemma 4.1. *We assume that the derivative of the nonlinear function f is bounded by a constant M; then there exists a norm, such that the discrete coarse propagator G satisfies*

$$\|G(T_{n+1}, T_n, U) - G(T_{n+1}, T_n, V)\| \leq (1 + C_2\Delta T)\|U - V\|,$$

where U, V are vectors in $\mathbb{R}^{J-1}$, and C_2 is a positive constant.

Proof. We only prove the result for the case $q = 1$, and we have $\Delta t = \Delta T$. From scheme (18), we have

$$\frac{1}{\Delta T}W_p - \frac{1}{\Delta T}W_{p-1} = \frac{1}{\Delta x^2}\tilde{A}W_p + f(W_{p-1}),$$

where

$$\tilde{A} = \begin{pmatrix} -2 & 1 & & \mathbf{0} \\ 1 & -2 & \ddots & \\ & \ddots & \ddots & 1 \\ \mathbf{0} & & 1 & -2 \end{pmatrix} \in \mathbb{R}^{(J-1)\times(J-1)}. \tag{20}$$

Then

$$W_p = \left(I - \frac{\Delta T}{\Delta x^2}\tilde{A}\right)^{-1} W_{p-1} + \Delta T \left(I - \frac{\Delta T}{\Delta x^2}\tilde{A}\right)^{-1} f(W_{p-1}),$$

where I is an identity matrix. The coarse propagator can be written as

$$G(T_{n+1}, T_n, U) = \left(I - \frac{\Delta T}{\Delta x^2}\tilde{A}\right)^{-1} U + \Delta T \left(I - \frac{\Delta T}{\Delta x^2}\tilde{A}\right)^{-1} f(U).$$

We notice that $\tilde{A}$ is a symmetric negative definite matrix. There exists an orthogonal matrix S, such that

$$\tilde{A} = S^{\mathrm{T}}\tilde{\Lambda}S,$$

where S^{T} denotes the transpose of the matrix S,

$$\tilde{\Lambda} = \begin{pmatrix} \lambda_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \lambda_{J-1} \end{pmatrix} \in \mathbb{R}^{(J-1)\times(J-1)},$$

and $\lambda_1, \ldots, \lambda_{J-1}$ are the eigenvalues of $\tilde{A}$, so $\lambda_i < 0$ for $i = 1, \ldots, J-1$. Furthermore, we have

$$\left(I - \frac{\Delta T}{\Delta x^2}\tilde{A}\right)^{-1} = S\left(I - \frac{\Delta T}{\Delta x^2}\tilde{\Lambda}\right)^{-1} S^{\mathrm{T}} = S \begin{pmatrix} \left(1 - \frac{\Delta T}{\Delta x^2}\lambda_1\right)^{-1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \left(1 - \frac{\Delta T}{\Delta x^2}\lambda_{J-1}\right)^{-1} \end{pmatrix} S^{\mathrm{T}}.$$

Obviously,

$$0 < \left(1 - \frac{\Delta T}{\Delta x^2}\lambda_i\right)^{-1} < 1,$$

for $i = 1, \ldots, J-1$. Therefore, all the eigenvalues of the matrix $(I - \frac{\Delta T}{\Delta x^2}\tilde{A})^{-1}$ are less than 1. Then, there exists a norm $\|\cdot\|$, such that $\|(I - \frac{\Delta T}{\Delta x^2}\tilde{A})^{-1}\| < 1$, and

$$\begin{aligned} \|G(T_{n+1}, T_n, U) - G(T_{n+1}, T_n, V)\| &\leq \left\|\left(I - \frac{\Delta T}{\Delta x^2}\tilde{A}\right)^{-1}\right\| [(1 + M\Delta T)\|U - V\|] \\ &\leq (1 + M\Delta T)\|U - V\|. \end{aligned}$$

The result follows. □

In fact, the result in Lemma 4.1 also holds for $q \geq 2$. For example, when $q = 2$ we have $\Delta T = 2\Delta t$. Denote

$$\tilde{G}(U) = \left(I - \frac{\Delta t}{\Delta x^2}\tilde{A}\right)^{-1} U + \Delta t \left(I - \frac{\Delta t}{\Delta x^2}\tilde{A}\right)^{-1} f(U),$$

and we have

$$\|G(T_{n+1}, T_n, U) - G(T_{n+1}, T_n, V)\| \leq \|\tilde{G}(\tilde{G}(U)) - \tilde{G}(\tilde{G}(V))\| \leq (1 + M\Delta t)\|\tilde{G}(U) - \tilde{G}(V)\| \leq (1 + M\Delta t)^2\|U - V\|.$$

There exists a constant C_2, such that $(1 + M\Delta t)^2 \leq 1 + C_2\Delta T$ for small ΔT.

Lemma 4.2. *Let $S_{\delta t}(T_{n+1}, T_n, U)$ be the approximation at T_{n+1} generated by scheme (17), with $V_0 = U$, then*

$$S_{\delta t}(T_{n+1}, T_n, U) - G(T_{n+1}, T_n, U) = c_2(U)\Delta T^2 + c_3(U)\Delta T^3 + \cdots,$$

where $U \in \mathbb{R}^{J-1}$, and $c_j(U)$ are continuously differentiable, for $j = 2, 3, \ldots$.

Proof. Let $S(T_{n+1}, T_n, U)$ be the true solution at T_{n+1}, with the initial value U at T_n, then

$$\begin{aligned} S_{\delta t}(T_{n+1}, T_n, U) - G(T_{n+1}, T_n, U) &= S_{\delta t}(T_{n+1}, T_n, U) - S(T_{n+1}, T_n, U) + S(T_{n+1}, T_n, U) - G(T_{n+1}, T_n, U) \\ &= \bar{c}_2(U)\Delta T^2 + \bar{c}_3(U)\Delta T^3 + \cdots + \tilde{c}_2(U)\Delta T^2 + \tilde{c}_3(U)\Delta T^3 + \cdots \\ &= c_2(U)\Delta T^2 + c_3(U)\Delta T^3 + \cdots. \end{aligned}$$

This completes the proof. □

The error estimation for the discrete WR algorithm (16) on the time slice $[T_n, T_{n+1}]$ can be described by the following theorem.

Theorem 4.1. *For system (6) defined on the time slice $[T_n, T_{n+1}]$ with the initial function $h(x)$ at T_n, we assume that the derivative of the nonlinear function f is bounded by a constant M, and denote the discrete sampling of $h(x)$ by $h_\Delta = [h(\Delta x), h(2\Delta x), \ldots, h((J-1)\Delta x)]^{\mathrm{T}}$. Then there exists a norm, such that the difference between $DW_i(T_{n+1}, T_n, h_\Delta)$ generated by scheme (16), and $S_{\delta t}(T_{n+1}, T_n, h_\Delta)$ generated by scheme (17), can be bounded by*

$$\|DW_i(T_{n+1}, T_n, h_\Delta) - S_{\delta t}(T_{n+1}, T_n, h_\Delta)\| \leq \bar{C}M^i\delta t^i\Delta T\frac{(i + \bar{q} - 1)(i + \bar{q} - 2)\cdots(i + 1)}{(\bar{q} - 1)!},$$

where $\bar{C}$ is a constant.

Proof. Define $\epsilon_m^{(i)} = V_m^{(i)} - V_m$, for $m = 0, 1, \ldots, \bar{q}$, $i = 0, 1, \ldots$. From (16) and (17), we have

$$\frac{1}{\delta t}\epsilon_m^{(i+1)} - \frac{1}{\delta t}\epsilon_{m-1}^{(i+1)} = \frac{1}{\Delta x^2}\tilde{A}\epsilon_m^{(i+1)} + f(V_m^{(i)}) - f(V_m),$$

where the matrix $\tilde{A}$ is defined in (20), and satisfies $\|(I - \frac{\delta t}{\Delta x^2}\tilde{A})^{-1}\| < 1$. Because

$$\epsilon_m^{(i+1)} = \left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1}\epsilon_{m-1}^{(i+1)} + \delta t\left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1}[f(V_m^{(i)}) - f(V_m)],$$

then

$$\begin{aligned} \|\epsilon_m^{(i+1)}\| &\leq \left\|\left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1}\right\|\|\epsilon_{m-1}^{(i+1)}\| + M\delta t\left\|\left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1}\right\|\|\epsilon_m^{(i)}\| \\ &\leq \|\epsilon_{m-1}^{(i+1)}\| + M\delta t\|\epsilon_m^{(i)}\|, \end{aligned}$$

with the estimation for the initial guess

$$\|\epsilon_m^{(0)}\| = \|h_\Delta - V_m\| \leq \bar{C}\Delta T.$$

We consider the recurrence relations

$$\bar{e}_m^{(i+1)} = \bar{e}_{m-1}^{(i+1)} + M\delta t\bar{e}_m^{(i)}, \qquad \bar{e}_m^{(0)} = \|\epsilon_m^{(0)}\|. \tag{21}$$

It is easy to check that the quantity $\bar{e}_m^{(i+1)}$ is an upper bound on $\|\epsilon_m^{(i+1)}\|$. The relations (21) can be rewritten as

$$\begin{pmatrix} \bar{e}_1^{(i+1)} \\ \bar{e}_2^{(i+1)} \\ \vdots \\ \bar{e}_{\bar{q}}^{(i+1)} \end{pmatrix} = \begin{pmatrix} 0 & & & \mathbf{0} \\ 1 & 0 & & \\ & \ddots & \ddots & \\ \mathbf{0} & & 1 & 0 \end{pmatrix}\begin{pmatrix} \bar{e}_1^{(i+1)} \\ \bar{e}_2^{(i+1)} \\ \vdots \\ \bar{e}_{\bar{q}}^{(i+1)} \end{pmatrix} + M\delta t\begin{pmatrix} \bar{e}_1^{(i)} \\ \bar{e}_2^{(i)} \\ \vdots \\ \bar{e}_{\bar{q}}^{(i)} \end{pmatrix}. \tag{22}$$

We denote $\vec{E}^{(i)} = (\bar{e}_1^{(i)}, \ldots, \bar{e}_{\bar{q}}^{(i)})^{\mathrm{T}}$, and

$$D = \begin{pmatrix} 0 & & & \mathbf{0} \\ 1 & 0 & & \\ & \ddots & \ddots & \\ \mathbf{0} & & 1 & 0 \end{pmatrix}_{\bar{q}\times\bar{q}},$$

then Eq. (22) can be rewritten as

$$\vec{E}^{(i+1)} = D\vec{E}^{(i+1)} + M\delta t\vec{E}^{(i)}.$$

Solving for $\vec{E}^{(i)}$, we obtain

$$\vec{E}^{(i)} = (M\delta t)^i (I - D)^{-i}\vec{E}^{(0)},$$

where I is an identity matrix of size $\bar{q} \times \bar{q}$, and

$$(I - D)^{-1} = \begin{pmatrix} 1 & & & \mathbf{0} \\ 1 & 1 & & \\ \vdots & \ddots & \ddots & \\ 1 & \cdots & 1 & 1 \end{pmatrix}, \qquad (I - D)^{-i} = \begin{pmatrix} 1 & & & & \mathbf{0} \\ C_i^1 & 1 & & & \\ C_{i+1}^2 & C_i^1 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \\ C_{i+\bar{q}-2}^{\bar{q}-1} & \cdots & C_{i+1}^2 & C_i^1 & 1 \end{pmatrix},$$

with

$$C_i^j = \binom{i}{j}. \tag{23}$$

Obviously, the last entry $\bar{e}_{\bar{q}}^{(i)}$ of $\vec{E}^{(i)}$ satisfies

$$\begin{aligned} \bar{e}_{\bar{q}}^{(i)} &\le (M\delta t)^i (1 + C_i^1 + C_{i+1}^2 + \cdots + C_{i+\bar{q}-2}^{\bar{q}-1}) \max_{1\le j\le\bar{q}} \bar{e}_j^{(0)} \\ &= (M\delta t)^i C_{i+\bar{q}-1}^{\bar{q}-1} \max_{1\le j\le\bar{q}} \|h_\Delta - V_j\| \\ &= \bar{C}\Delta T (M\delta t)^i C_{i+\bar{q}-1}^{\bar{q}-1} \\ &\le \bar{C} M^i \delta t^i \Delta T \frac{(i+\bar{q}-1)(i+\bar{q}-2)\cdots(i+1)}{(\bar{q}-1)!}. \end{aligned}$$

The result follows. □

Based on Lemmas 4.1 and 4.2 and Theorem 4.1, the error estimation for the discrete parareal WR algorithm (19) is included in the following theorem.

Theorem 4.2. *For system (6), we assume that the derivative of the nonlinear function f is bounded by a constant M; then the discrete parareal WR algorithm (19) converges to the fine approximation of system (6) generated by (17), with the error estimation*

$$\|U_n - U_n^k\| \le \frac{C_3 e^{C_2 T_{n-k-1}}}{C_1} \frac{(C_1 T_n \Delta T)^{k+1}}{(k+1)!} + \bar{C}\Delta T^k \sum_{i=0}^{k-1} e^{C_2 T_{n-i-1}} \frac{\prod_{j=1}^{\bar{q}-1}(k-i+j)}{\bar{q}^{k-i}(\bar{q}-1)!} \frac{C_1^i M^{k-i} T_n^{i+1}}{(i+1)!} \tag{24}$$

where C_1, C_2 are from Lemmas 4.1 and 4.2, and C_3, $\bar{C}$ are constants.

Proof. From scheme (19) we have

$$\begin{aligned} U_{n+1} - U_{n+1}^{k+1} &= S_{\delta t}(T_{n+1}, T_n, U_n) - G(T_{n+1}, T_n, U_n^{k+1}) - DW_{k+1}(T_{n+1}, T_n, U_n^k) + G(T_{n+1}, T_n, U_n^k) \\ &= [S_{\delta t}(T_{n+1}, T_n, U_n) - G(T_{n+1}, T_n, U_n) + G(T_{n+1}, T_n, U_n^k) - S_{\delta t}(T_{n+1}, T_n, U_n^k)] \\ &\quad + [S_{\delta t}(T_{n+1}, T_n, U_n^k) - DW_{k+1}(T_{n+1}, T_n, U_n^k)] + [G(T_{n+1}, T_n, U_n) - G(T_{n+1}, T_n, U_n^{k+1})]. \end{aligned}$$

By Lemmas 4.1 and 4.2, we have

$$\|U_{n+1} - U_{n+1}^{k+1}\| \le C_1 \Delta T^2 \|U_n - U_n^k\| + \varepsilon(k+1)\Delta T + (1 + C_2\Delta T)\|U_n - U_n^{k+1}\|,$$

where

$$\varepsilon(k) = \bar{C}M^k \delta t^k \frac{(k+\bar{q}-1)(k+\bar{q}-2)\cdots(k+1)}{(\bar{q}-1)!}.$$

Similar to the proof of Theorem 3.1, we have

$$\|U_n - U_n^k\| \le \frac{C_3 e^{C_2 T_{n-k-1}}}{C_1} \frac{(C_1 T_n \Delta T)^{k+1}}{(k+1)!} + \bar{C}\Delta T^k \sum_{i=0}^{k-1} e^{C_2 T_{n-i-1}} \frac{\prod_{j=1}^{\bar{q}-1}(k-i+j)}{\bar{q}^{k-i}(\bar{q}-1)!} \frac{C_1^i M^{k-i} T_n^{i+1}}{(i+1)!}.$$

This completes the proof. □

5. The balance of the two kinds of iterations

The parareal WR algorithm involves two iterative processes, parareal and WR. The two processes usually converge at different rates, the parallel efficiency will be slowed down by the worse one. In this section we show a simple strategy for choosing the time step of the coarse propagator to balance the two iterative processes, by the following linear ODEs,

$$\begin{cases} \dfrac{du(t)}{dt} = \dfrac{1}{\Delta x^2}\tilde{A}u(t) + Lu(t), \quad 0 < t < T, \\ u(0) = U_0 \in \mathbb{R}^{J-1}, \end{cases} \tag{25}$$

where the matrix $\tilde{A}$ is defined in (20), and L is a constant.

We divide $[0, T]$ into N time slices with a uniform length ΔT, and on each time slice $[T_n, T_{n+1}]$ perform the following WR scheme,

$$\begin{cases} \dfrac{du_n^{(i+1)}(t)}{dt} = \dfrac{1}{\Delta x^2}\tilde{A}u_n^{(i+1)}(t) + Lu_n^{(i)}(t), \quad T_n < t < T_{n+1}, \\ u_n^{(i+1)}(0) = U_n. \end{cases} \tag{26}$$

The fine propagator is generated by scheme (17) for system (26), and the coarse propagator is generated by scheme (18) for system (25).

Similar to the proof of Theorem 3.1, we can obtain the recurrence relations as follows

$$\hat{e}_{n+1}^{k+1} = \alpha\hat{e}_n^k + \beta\hat{e}_n^{k+1} + \varepsilon_n(k+1)\Delta T, \qquad \hat{e}_{n+1}^0 = \gamma + \beta\hat{e}_n^0, \tag{27}$$

where $\hat{e}_n^k$ is an upper bound on the error at T_n caused by the parareal WR algorithm for system (25), and $\varepsilon_n(k+1)$ denotes the upper bound on the error caused by $(k+1)$ iterations of discrete WR on $[T_n, T_{n+1}]$. The relations (27) have the compact form

$$\begin{pmatrix} \hat{e}_1^{k+1} \\ \hat{e}_2^{k+1} \\ \vdots \\ \hat{e}_N^{k+1} \end{pmatrix} = \begin{pmatrix} 1 & & & \mathbf{0} \\ -\beta & 1 & & \\ & \ddots & \ddots & \\ \mathbf{0} & & -\beta & 1 \end{pmatrix}^{-1} \left[\begin{pmatrix} 0 & & & \mathbf{0} \\ \alpha & 0 & & \\ & \ddots & \ddots & \\ \mathbf{0} & & \alpha & 0 \end{pmatrix} \begin{pmatrix} \hat{e}_1^k \\ \hat{e}_2^k \\ \vdots \\ \hat{e}_N^k \end{pmatrix} + \begin{pmatrix} \varepsilon_1(k+1) \\ \varepsilon_2(k+1) \\ \vdots \\ \varepsilon_N(k+1) \end{pmatrix} \right]. \tag{28}$$

We first investigate the convergence factor of the discrete WR for system (25) on $[T_n, T_{n+1}]$, which is generated by

$$\frac{V_{m+1}^{(i+1)} - V_m^{(i+1)}}{\delta t} = \frac{1}{\Delta x^2}\tilde{A}V_{m+1}^{(i+1)} + LV_{m+1}^{(i)}, \quad m = 0, 1, \ldots, \bar{q}-1, \tag{29}$$

and the discrete scheme for system (25) on the same fine time grid is

$$\frac{V_{m+1} - V_m}{\delta t} = \frac{1}{\Delta x^2}\tilde{A}V_{m+1} + LV_{m+1}. \tag{30}$$

We define $\epsilon_{m+1}^{(i)} = V_{m+1}^{(i)} - V_{m+1}$. From (29) and (30) we have

$$\epsilon_{m+1}^{(i+1)} = \left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1} \epsilon_m^{(i+1)} + L\delta t \left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1} \epsilon_{m+1}^{(i)},$$

and

$$\|\epsilon_{m+1}^{(i+1)}\|_\infty \le \left\| \left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1} \right\|_\infty \|\epsilon_m^{(i+1)}\|_\infty + L\delta t \left\| \left(I - \frac{\delta t}{\Delta x^2}\tilde{A}\right)^{-1} \right\|_\infty \|\epsilon_{m+1}^{(i)}\|_\infty.$$

We denote $a_0 = \|(I - \frac{\delta t}{\Delta x^2}\tilde{A})^{-1}\|_\infty$. In fact, a large number of numerical tests support $a_0 \le 1$, where most of the values of a_0 are very close to 1 or even equal to 1. Then we have

$$\|\epsilon_{m+1}^{(i+1)}\|_\infty \le \|\epsilon_m^{(i+1)}\|_\infty + L\delta t\|\epsilon_{m+1}^{(i)}\|_\infty.$$

We employ the following recurrence relations

$$\tilde{e}_{m+1}^{(i+1)} = \tilde{e}_m^{(i+1)} + L\delta t\tilde{e}_{m+1}^{(i)}, \qquad \tilde{e}_m^{(0)} = \|\epsilon_m^{(0)}\|_\infty,$$

and their compact form

$$\begin{pmatrix} \tilde{e}_1^{(i+1)} \\ \tilde{e}_2^{(i+1)} \\ \vdots \\ \tilde{e}_{\bar{q}}^{(i+1)} \end{pmatrix} = \begin{pmatrix} 0 & & & \mathbf{0} \\ 1 & 0 & & \\ & \ddots & \ddots & \\ \mathbf{0} & & 1 & 0 \end{pmatrix} \begin{pmatrix} \tilde{e}_1^{(i+1)} \\ \tilde{e}_2^{(i+1)} \\ \vdots \\ \tilde{e}_{\bar{q}}^{(i+1)} \end{pmatrix} + L\delta t \begin{pmatrix} \tilde{e}_1^{(i)} \\ \tilde{e}_2^{(i)} \\ \vdots \\ \tilde{e}_{\bar{q}}^{(i)} \end{pmatrix},$$

where $\tilde{e}_m^{(i)}$ is an upper bound for $\|\epsilon_m^{(i+1)}\|_\infty$. Furthermore, we can obtain

$$\begin{pmatrix} \tilde{e}_1^{(i)} \\ \tilde{e}_2^{(i)} \\ \vdots \\ \vdots \\ \tilde{e}_{\bar{q}}^{(i)} \end{pmatrix} = (L\delta t)^i \begin{pmatrix} 1 & & & & \mathbf{0} \\ C_i^1 & 1 & & & \\ C_{i+1}^2 & C_i^1 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \\ C_{i+\bar{q}-2}^{\bar{q}-1} & \cdots & C_{i+1}^2 & C_i^1 & 1 \end{pmatrix} \begin{pmatrix} \tilde{e}_1^{(0)} \\ \tilde{e}_2^{(0)} \\ \vdots \\ \vdots \\ \tilde{e}_{\bar{q}}^{(0)} \end{pmatrix},$$

where the C_i^j are defined according to (23). From the above equations we have

$$\tilde{e}_{\bar{q}}^{(i)} \le (L\delta t)^i C_{i+\bar{q}-1}^{\bar{q}-1} \max_{1\le j\le \bar{q}} \tilde{e}_{\bar{q}}^{(0)},$$

$$\tilde{e}_{\bar{q}}^{(i+1)} \le (L\delta t)^{i+1} C_{i+\bar{q}}^{\bar{q}-1} \max_{1\le j\le \bar{q}} \tilde{e}_{\bar{q}}^{(0)} = L\delta t \frac{i+\bar{q}}{i+1} (L\delta t)^i C_{i+\bar{q}-1}^{\bar{q}-1} \max_{1\le j\le \bar{q}} \tilde{e}_{\bar{q}}^{(0)}.$$

Therefore, the relationship between $\varepsilon(k+1)$ and $\varepsilon(k)$ in (28) can be described approximately by

$$\varepsilon(k+1) \approx L\delta t \frac{k+\bar{q}}{k+1}\varepsilon(k). \tag{31}$$

Now, we consider the factor α in (28), which describes the truncation error of the coarse propagator. The coarse propagation for system (25) on each time slice $[T_n, T_{n+1}]$ is generated by

$$\frac{W_{p+1} - W_p}{\Delta t} = \frac{1}{\Delta x^2}\tilde{A}W_{p+1} + LW_p, \quad p = 0, 1, \ldots, q-1,$$

which leads to

$$W_{p+1} = (1 + L\Delta t)\left(I - \frac{\Delta t}{\Delta x^2}\tilde{A}\right)^{-1} W_p.$$

Then the coarse propagation can be written as

$$G(T_{n+1}, T_n, U) = \left[(1 + L\Delta t)\left(I - \frac{\Delta t}{\Delta x^2}\tilde{A}\right)^{-1}\right]^q U.$$

Likewise, the fine discretization of system (25) is

$$\frac{V_{m+1} - V_m}{\delta t} = \frac{1}{\Delta x^2}\tilde{A}V_{m+1} + LV_{m+1}, \quad m = 0, 1, \ldots, \bar{q}-1,$$

and the discrete counterpart of the true solution at T_{n+1} can be written as

$$S_{\delta t}(T_{n+1}, T_n, U) = \left[(1 - L\delta t)I - \frac{\delta t}{\Delta x^2}\tilde{A}\right]^{-\bar{q}} U.$$

Therefore, the factor α can be estimated by

$$\left\| \left[(1 - L\delta t)I - \frac{\delta t}{\Delta x^2}\tilde{A}\right]^{-\bar{q}} - \left[(1 + L\Delta t)\left(I - \frac{\Delta t}{\Delta x^2}\tilde{A}\right)^{-1}\right]^q \right\|_\infty.$$

According to the relations (28), the parareal iteration and the WR iteration can be balanced, if we make the factor α equal to the factor $L\delta t\frac{k+\bar{q}}{k+1}$. It means that we can find a proper Δt by the following equation,

$$\left\|\left[(1-L\delta t)I-\frac{\delta t}{\Delta x^2}\tilde{A}\right]^{-\bar{q}}-\left[(1+L\Delta t)\left(I-\frac{\Delta t}{\Delta x^2}\tilde{A}\right)^{-1}\right]^{q}\right\|_{\infty}=L\delta t\frac{k+\bar{q}}{k+1}. \tag{32}$$

In practice, we have several other considerations when choosing the values for Δt (or for q). For example, too small values of Δt will introduce too much computational cost for the coarse propagations, while too big values of Δt will lead to many parareal iterations. Therefore, the value for Δt by (32) can be seen as a feasible choice, and we can adjust the value properly, if the Δt is too big or too small.

6. Parallel efficiency

There are two kinds of parallel fashions for the classical parareal algorithm, serial parareal and pipeline parareal; see [19]. Similar to the analysis for parallel efficiency shown in [19,20], we assume that the cost of information transmission and data storage are negligible, and adopt the following notations.

- T is the length of time domain.
- ΔT is the length of each time slice.
- Δt is the time increment for the coarse propagator.
- δt is the time increment for the fine propagator.
- N is the number of time slices on $[0, T]$, i.e., $T = N\Delta T$.
- q is the number of time steps for each coarse propagation.
- $\bar{q}$ is the number of time steps for each fine propagation.
- τ_G is the cost of the numerical method at each time step in the coarse propagation.
- τ_F is the cost of the numerical method at each time step in the fine propagation.
- k is the iteration number for the parareal WR algorithm.

First, we suppose that enough processors are available, and each fine propagation in the kth parareal iteration is implemented by k processors. The total cost for the initial step and k iterations of parareal WR algorithm in the pipelined fashion is

$$Nq\tau_G+(q\tau_G+\bar{q}\tau_F)+(q\tau_G+(\bar{q}+1)\tau_F)+\cdots+(q\tau_G+(\bar{q}+k-1)\tau_F)=(k+N)q\tau_G+k\bar{q}\tau_F+\frac{k(k-1)}{2}\tau_F.$$

The time cost of applying windowing WR serially for system (6) is $kN\bar{q}\tau_F$. The speedup for parareal WR is

$$S=\frac{kN\bar{q}\tau_F}{(k+N)q\tau_G+k\bar{q}\tau_F+\frac{k(k-1)}{2}\tau_F}.$$

If we further assume that $\tau_G=\tau_F$, the speedup will be

$$S=\frac{kN\bar{q}}{(k+N)q+k\bar{q}+\frac{k(k-1)}{2}},$$

and the parallel efficiency is

$$E=\frac{S}{kN}=\frac{\bar{q}}{(k+N)q+k\bar{q}+\frac{k(k-1)}{2}}, \tag{33}$$

which is almost the same as that of the classical parareal algorithm. However, many processors do not work in the first several parareal iterations.

Now, we suppose that each fine propagation in the kth parareal iteration is implemented by one processor. The total cost for the initial step and k iterations of parareal WR algorithm is

$$Nq\tau_G+(q\tau_G+\bar{q}\tau_F)+(q\tau_G+2\bar{q}\tau_F)+\cdots+(q\tau_G+k\bar{q}\tau_F)=(k+N)q\tau_G+\frac{k(k+1)}{2}\bar{q}\tau_F.$$

The corresponding speedup for parareal WR with the assumption $\tau_G=\tau_F$ is

$$\bar{S}=\frac{kN\bar{q}}{(k+N)q+\frac{k(k+1)}{2}\bar{q}},$$

and the parallel efficiency is

$$\bar{E}=\frac{\bar{S}}{N}=\frac{\bar{q}}{\frac{(k+N)}{k}q+\frac{(k+1)}{2}\bar{q}}, \tag{34}$$

which is much better than the efficiency in (33).

Table 1
The errors and the running time of the parareal WR algorithm and the corresponding WR algorithm for system (35), with $q = 2, \bar{q} = 1000$ and $J = 100$.

k		$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Time of parareal WR (s)	3CPU	14.52	42.44	84.44	140.32	210.28	294.57
	6CPU	5.73	17.03	33.87	56.25	84.25	117.84
	11CPU	2.98	8.77	17.45	28.77	43.10	60.04
	21CPU	1.72	6.42	8.74	14.54	21.74	30.12
	41CPU	0.98	2.42	4.51	7.38	10.99	15.24
Parareal WR error		1.88e−002	5.70e−004	2.52e−005	1.17e−006	1.48e−007	2.03e−008
k		$k = 7$	$k = 8$	$k = 9$	$k = 10$	$k = 11$	$k = 12$
Time of parareal WR (s)	3CPU	392.14	504.18	630.24	771.26	924.65	1091.13
	6CPU	157.13	201.65	251.99	307.98	369.32	436.06
	11CPU	80.01	102.73	129.32	156.81	188.12	223.73
	21CPU	40.04	51.37	64.54	78.29	93.84	110.84
	41CPU	20.39	25.94	32.42	39.33	47.20	55.64
Parareal WR error		2.74e−009	3.87e−010	5.56e−011	7.86e−012	1.10e−012	1.63e−013
k		$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Time of WR (s)		28.60	56.51	85.05	113.42	141.14	169.40
WR error		4.41e−003	3.67e−005	3.02e−007	2.43e−009	1.91e−011	1.43e−013

For different arrangements for the processors employed to implement the parareal WR algorithm, the resulting parallel efficiencies are usually greater than E, and less than E. Moreover, we will see from the numerical results in the next section that, the parareal WR algorithm costs much less running time than the corresponding WR algorithm on massively parallel computers.

7. Numerical experiments

In this section, we take the following one-dimensional parabolic PDE as an example to carry out the parareal WR algorithm,

$$\begin{cases} \dfrac{\partial u}{\partial t} - \dfrac{\partial^2 u}{\partial x^2} = e^{\frac{u}{10+10u}}, \quad 0 < x < 1,\ 0 < t < 1, \\ u(0, t) = u(1, t) = 0, \quad 0 \le t \le 1, \\ u(x, 0) = \sin \pi x, \quad 0 \le x \le 1. \end{cases} \tag{35}$$

The time domain $[0, 1]$ is divided into 40 time slices. The following scheme for WR is employed on each time slice,

$$\begin{cases} \dfrac{\partial u^{(k+1)}}{\partial t} - \dfrac{\partial^2 u^{(k+1)}}{\partial x^2} = e^{\frac{u^{(k)}}{10+10u^{(k)}}}, \quad 0 < x < 1,\ T_n < t < T_{n+1}, \\ u^{(k+1)}(0, t) = u^{(k+1)}(1, t) = 0, \quad T_n \le t \le T_{n+1}, \\ u^{(k+1)}(x, 0) = U_n^k(x), \quad 0 \le x \le 1. \end{cases} \tag{36}$$

For the coarse propagator, we take scheme (18) for system (35) on each time slice, with $q = 2$ and $J = 100$. For the fine propagator, we take scheme (17) for system (36), with $\bar{q} = 1000$ and $J = 100$. For the referee solution, we take scheme (17), with enough WR iterations and the parameters $\bar{q} = 1000$ and $J = 100$, for system (36) on one time slice after another sequentially. Let $U_{n,j}^k$ be the approximation by the parareal WR algorithm to $u(j\Delta x, n\Delta T)$, and $U_{n,j}$ be the referee solution at the point $(j\Delta x, n\Delta T)$. The resulting error is measured by $\max_{1\le n\le 40, 1\le j\le 99} |U_{n,j}^k - U_{n,j}|$. All algorithms are implemented in C and MPI on TYAN FX71 AMD Opteron. This multi-core computer has 6 chips, and each chip has two quad-core AMD Opteron 2350, which run at frequencies exceeding 2000 MHz. It means that up to 48 cores are available during computation.

In this paper, we concentrate on the parallelism on time domain. In detail, we employ N_p processors, including one master processor and $N_p - 1$ slave processors, to implement the parareal WR algorithm for system (35). In each parareal iteration, the G propagations are carried out by the master processor, and the W_k propagations including k iterations of WR are computed by the slave processors. For example, if $N_p = 21$, each processor computes two W_k propagations during each parareal iteration; if $N_p = 41$, each processor computes one W_k propagation during each parareal iteration.

The error and the running time, for the parareal WR algorithm with different number of processors, are recorded in Table 1. Fig. 1 displays the convergence behavior of the parareal WR algorithm and the corresponding theoretical upper bound on errors, where the theoretical bound is shown in (24), with the estimated parameters $C_1 = 50.88$, $C_2 = 0.1$, $C_3 = 14.20, \bar{C} = 7.46, M = 0.1$. In order to show the effectiveness of the parareal WR algorithm, we perform the windowing WR algorithm with the same parameters as in the W_k propagation, and the result are also shown in Table 1. It can be found that the parareal WR algorithm with such parameters converges more slowly than the corresponding WR algorithm. Next, we show two strategies to improve the parallel efficiency of the parareal WR algorithm.

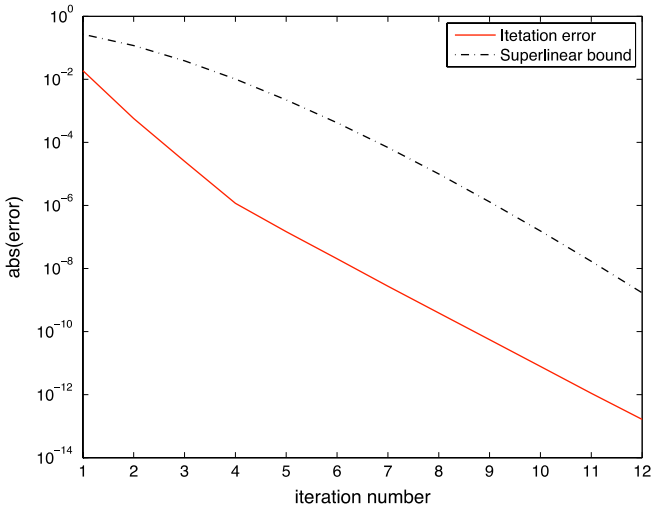

Fig. 1. The convergence of the parareal WR algorithm for system (35): the observed error and theoretical superlinear bound, with $q = 2$, $\bar{q} = 1000$ and $J = 100$.

Table 2
The error and the running time of the parareal WR algorithm, with $q = 2$, $\bar{q} = 1000$ and $J = 100$, where $1 + \lfloor k/2 \rfloor$ iterations of WR are carried out in the kth parareal iteration.

k		$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Time	3CPU	14.17	42.42	70.53	112.69	154.72	210.70
of	6CPU	5.73	17.04	28.29	45.14	61.96	84.31
parareal	11CPU	2.98	8.80	14.55	23.12	31.93	43.07
WR	21CPU	1.56	4.46	7.35	11.57	15.87	21.46
(s)	41CPU	0.92	2.38	3.84	6.08	8.21	11.10
Parareal WR error		1.88e−002	5.70e−004	2.52e−005	1.16e−006	1.49e−007	2.05e−008
k		$k = 7$	$k = 8$	$k = 9$	$k = 10$	$k = 11$	$k = 12$
Time	3CPU	266.74	336.66	406.94	490.63	574.68	672.60
of	6CPU	106.71	134.85	162.68	196.27	229.88	269.03
parareal	11CPU	54.58	68.80	83.00	100.23	117.16	137.14
WR	21CPU	27.15	34.21	41.29	49.68	58.23	68.01
(s)	41CPU	14.01	17.62	21.21	25.52	29.82	34.83
Parareal WR error		2.75e−009	3.89e−010	5.59e−011	7.90e−012	1.11e−012	1.64e−013

Table 3
The error and the running time of the parareal WR algorithm, with $q = 2$, $\bar{q} = 1000$ and $J = 100$, where $1 + \lfloor k/3 \rfloor$ iterations of WR are carried out in the kth parareal iteration.

k		$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Time	3CPU	14.16	28.44	56.41	84.49	112.50	154.74
of	6CPU	5.72	11.46	22.68	33.89	45.10	61.89
parareal	11CPU	3.00	5.98	11.66	17.55	23.18	31.80
WR	21CPU	1.56	3.06	5.94	8.79	11.70	16.01
(s)	41CPU	0.84	1.64	3.17	4.60	6.09	8.32
Parareal WR error		1.88e−002	5.70e−004	1.99e−004	8.38e−006	4.22e−007	9.18e−008
k		$k = 7$	$k = 8$	$k = 9$	$k = 10$	$k = 11$	$k = 12$
Time	3CPU	196.52	238.70	294.36	350.24	406.51	476.02
of	6CPU	78.68	95.45	117.78	140.15	162.53	190.50
parareal	11CPU	40.44	48.97	60.37	71.92	83.34	97.60
WR	21CPU	20.35	24.62	30.53	35.98	42.01	48.83
(s)	41CPU	10.54	12.64	15.51	18.45	21.32	24.88
Parareal WR error		4.25e−009	3.39e−010	4.64e−011	5.24e−012	6.65e−013	1.57e−013

First, we reduce the iteration number of WR in each parareal iteration. For example, if we take $1 + \lfloor k/2 \rfloor$ WR iterations in the kth parareal iteration, where $\lfloor a \rfloor$ denotes the integer part of a, the resulting running time and the error are recorded in Table 2. We can see that, the tiny modification really saves much running time, and almost preserves the accuracy.

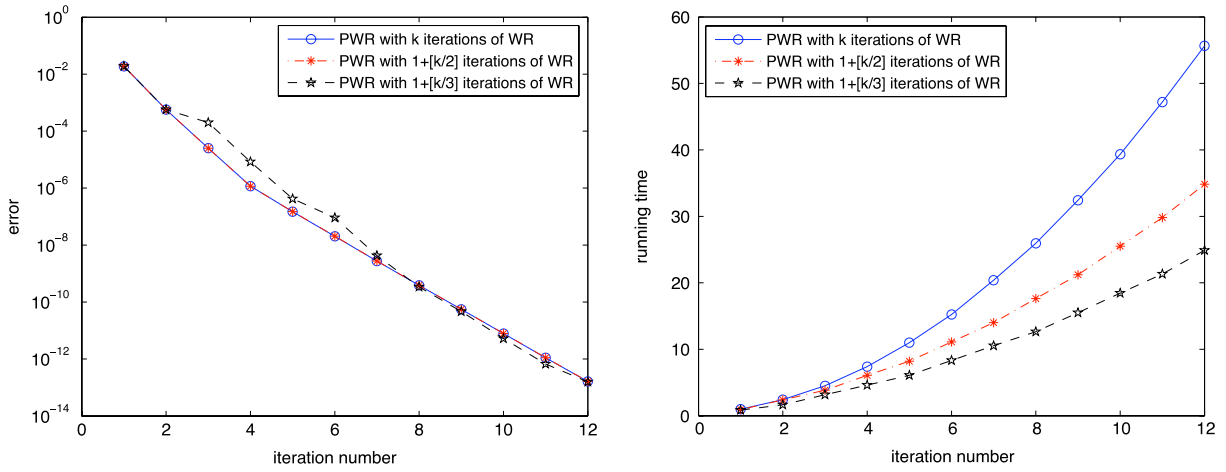

Fig. 2. Relationships between the errors and the iteration number of parareal WR algorithm (left). Relationships between the running time and the iteration number (right).

Table 4
The error and the running time of the parareal WR algorithm, with $q = 20$, $\bar{q} = 1000$ and $J = 100$, where k iterations of WR are carried out in the kth parareal iteration.

k		$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Time	3CPU	15.13	43.65	86.05	142.33	212.63	297.01
of	6CPU	6.72	18.45	35.70	58.59	87.01	121.05
parareal	11CPU	3.68	10.07	19.15	30.87	45.60	62.34
WR	21CPU	2.42	7.72	10.44	16.64	24.24	32.42
(s)	41CPU	1.68	3.72	6.21	9.48	13.49	17.54
Parareal WR error		1.93e−003	1.82e−004	6.86e−007	3.12e−009	2.41e−011	3.51e−013

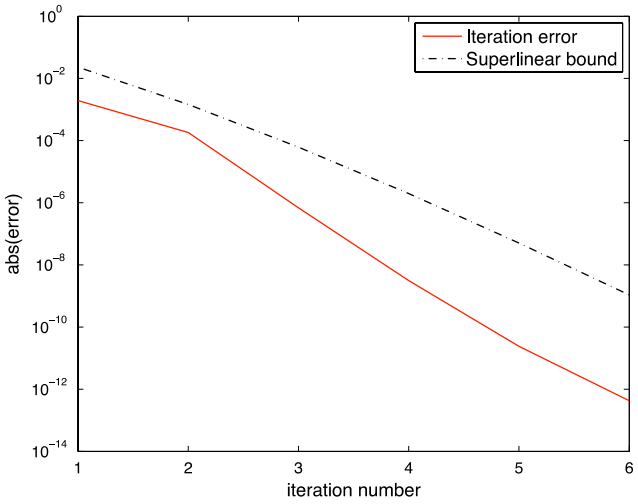

Fig. 3. The convergence of the parareal WR algorithm for system (35): the observed error and theoretical superlinear bound, with $q = 20$, $\bar{q} = 1000$ and $J = 100$.

Furthermore, if we take $1 + \lfloor k/3 \rfloor$ WR iterations in the kth parareal iteration, the running time and the error are shown in Table 3. The results for the three parareal WR algorithms with 41 processors are compared in Fig. 2.

Second, we can improve the convergence of the parareal process, by increasing the value of q. Let $q = 20$ and keep other parameters, the resulting errors and the running time are shown in Table 4. Fig. 3 displays the convergence behavior of the parareal WR algorithm and the corresponding theoretical upper bound (24) on errors, with the estimated parameters $C_1 = 5.14$, $C_2 = 0.1$, $C_3 = 1.45$, $\bar{C} = 7.46$, $M = 0.1$.

8. Conclusion and future work

The parareal and a kind of WR techniques have been combined to develop a new parallel algorithm, which could be carried out in parallel in two different directions. Sharp bounds on errors were presented, which indicate the approximately superlinear convergence for the new algorithm on bounded time domain. The convergence of the parareal WR algorithm on unbounded time domain has also been analyzed. In point of view of the parallel efficiency, the new approach was superior to the classical parareal algorithm. Meanwhile, we observed by numerical experiments that much less running time was needed on a massively parallel computer, than the corresponding WR algorithm, to achieve the desired accuracy. However, it would worth for us to further consider some correction methods to promote the efficiency on the combination of the two different parallel approaches.

Acknowledgments

The authors would like to thank the anonymous referees for their constructive criticisms and helpful comments to improve the paper significantly.

References

[1] E. Lelarasmee, A.E. Ruehli, A.L. Sangiovanni-Vincentelli, The waveform relaxation method for time-domain analysis of large scale integrated circuits, IEEE Transactions on Computer-Aided Design 1 (3) (1982) 131–145.
[2] Y.L. Jiang, A general approach to waveform relaxation solutions of nonlinear differential-algebraic equations: the continuous-time and discrete-time cases, IEEE Transactions on Circuits and Systems I - Regular Papers 51 (2004) 1770–1780.
[3] Y.L. Jiang, R.M.M. Chen, Computing periodic solutions of linear differential-algebraic equations by waveform relaxation, Mathematics of Computation 74 (2005) 781–804.
[4] Y.L. Jiang, R.M.M. Chen, O. Wing, Improving convergence performance of relaxation-based transient analysis by matrix splitting in circuit simulation, IEEE Transactions on Circuits and Systems I - Regular Papers 48 (2001) 769–780.
[5] Y.L. Jiang, O. Wing, A note on convergence conditions of waveform relaxation algorithms for nonlinear differential-algebraic equations, Applied Numerical Mathematics 36 (2001) 281–297.
[6] J. Liu, Y.L. Jiang, Waveform relaxation for reaction–diffusion equations, Journal of Computational and Applied Mathematics 235 (2011) 5040–5055.
[7] J.L. Lions, Y. Maday, G. Turinici, A "parareal" in time discretization of PDE's, Comptes Rendus del Academie des Sciences Serie I-Mathematique 332 (2001) 661–668.
[8] P. Amodio, L. Brugnano, Parallel solution in time of ODEs: some achievements and perspectives, Applied Numerical Mathematics 59 (2009) 424–435.
[9] P. Amodio, L. Brugnano, Parallel implementation of block boundary value methods for ODEs, Journal of Computational and Applied Mathematics 78 (1997) 197–211.
[10] P. Amodio, L. Brugnano, Parallel ODE solvers based on block BVMs, Advances in Computational Mathematics 7 (1997) 5–26.
[11] M.J. Gander, S. Vandewalle, Analysis of the parareal time-parallel time-integration method, SIAM Journal of Scientific Computing 29 (2007) 556–578.
[12] S. Vandewalle, Multilevel time-integration: analysis of the parareal algorithm, in: Numerical Analysis Seminar Series. Oxford Computing Laboratory, University of Oxford, New York, February 8, 2007.
[13] S. Vandewalle, M.J. Gander, The parareal algorithm in a historical perspective, in: 16th International Conference on Domain Decomposition Methods on Science and Engineering, New York, January 12–15, 2005.
[14] M.J. Gander, M. Petcu, Analysis of a Krylov subspace enhanced parareal algorithm for linear problems, ESAIM Proceedings 25 (2008) 114–129.
[15] G. Bal, Y. Maday, A "parareal" time discretization for non-linear PDE's with application to the pricing of an American put, in: Workshop on Domain Decomposition, Zurich, June 7–8, 2001.
[16] P.F. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the Navier–Stokes equations, in: 15th International Conference on Domain Decomposition Methods in Science and Engineering, Berlin, July 21–25, 2003.
[17] Y. Maday, J. Salomon, G. Turinici, Monotonic parareal control for quantum systems, SIAM Journal Numerical Analysis 45 (2007) 2468–2482.
[18] M.J. Gander, E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: 17th International Conference on Domain Decomposition Methods on Science and Engineering, St. Wolfgang/Strobl, July 3–7, 2006.
[19] M. Minion, A hybrid parareal spectral deferred correction method, Communications in Applied Mathematics and Computational Science 5 (2010) 265–301.
[20] G. Bal, Parallelization in time of (stochastic) ordinary differential equations, available: www.columbia.edu/~gb2030/PAPERS/paralleltime.pdf.